In Proceedings of IEEE INFOCOM'93

Rate-Controlled Static-Priority Queueing[*]

Hui Zhang and Domenico Ferrari
hzhang, ferrari@tenet.Berkeley.EDU
Computer Science Division
University of California at Berkeley
Berkeley, CA 94720

Abstract

We propose a new service discipline, called the Rate-Controlled Static-Priority (RCSP) queueing discipline, that can provide throughput, delay, delay jitter, and loss free guarantees in a connection-oriented packet-switching network. Previously proposed solutions are based on either a time-framing strategy, or a sorted priority queue mechanism. Time-framing schemes suffer from the dependencies they introduce between the queueing delay and the granularity of bandwidth allocation; sorted priority queues are more complex, and may be difficult to implement. The proposed RCSP queueing discipline avoids both time-framing and sorted priority queues; it achieves flexibility in the allocation of delay and bandwidth, as well as simplicity of implementation. The key idea is to separate rate-control and delay-control functions in the design of the server. Applying this separation of functions will result in a class of service disciplines, of which RCSP is an instance.

1 Introduction

Future high speed networks will have to support real-time communication services, which allow clients to transport information with performance guarantees expressed in terms of delay, delay jitter, throughput and loss rate bounds. It has been argued that a connection-oriented architecture, with explicit resource allocation and connection admission control, is needed to offer such a real-time service [9]. However, in a packet-switching network, packets from different connections will interact with each other at each switch; without proper control, these interactions may adversely affect the network performance experienced by clients. The service disciplines at the switching nodes, which control the order in which packets are serviced, determine how packets from different connections interact with each other.

The First-Come-First-Served (FCFS) discipline, which is used in most conventional networks, may be capable of providing some performance guarantees, but only under a number of constraints and with a very inefficient use of the network's resources [8]. New service disciplines that aim to provide different qualities of service to different connections have been proposed. These include Delay Earliest-Due-Date (Delay-EDD) [9], Virtual Clock [20], Fair Queueing [5] and its weighted version [15], Jitter Earliest-Due-Date (Jitter-EDD) [18], Stop-and-Go [10], and Hierarchical Round Robin (HRR) [12]. These solutions are based on either a time-framing strategy, or a sorted priority queue mechanism. Framing introduces dependencies between scheduling priority and bandwidth allocation granularity, so that connections with both low delay and low throughput requirements cannot be supported efficiently. A sorted priority queue has an $O(\log N)$ insertion operation [14], where N is the number of packets in the queue. This may not be feasible in a high-speed implementation. Also, in order to decouple scheduling priority and bandwidth allocation, a scheme based on a sorted priority queue requires a complicated schedulability test at connection establishment time [9, 19], which is not needed in time-framing-based schemes like Stop-and-Go and HRR.

In this paper, we present a new service discipline called *Rate-Controlled Static-Priority (RCSP) queueing* that overcomes these limitations. In particular, RCSP (i) provides delay, throughput and loss-free guarantees; (ii) introduces no coupling between delay and bandwidth allocation; (iii) is relatively easy to implement, since no sorted priority queue is needed; and (iv) requires relatively simple admission control tests.

RCSP can either be work-conserving[1] or non-work-conserving. The work-conserving version of RCSP utilizes

---

[*] This research was supported by the National Science Foundation and the Defense Advanced Research Projects Agency (DARPA) under Cooperative Agreement NCR-8919038 with the Corporation for National Research Initiatives, by AT&T Bell Laboratories, Hitachi, Ltd., Hitachi America, Ltd., Pacific Bell, the University of California under a MICRO grant, and the International Computer Science Institute. The views and conclusions contained in this document are those of the authors, and should not be interpreted as representing official policies, either expressed or implied, of the U.S. Government or any of the sponsoring organizations.

[1] In a work-conserving discipline, a server is never idle when there is a packet to transmit [19]

bandwidth more efficiently while the non-work-conserving version of RCSP has the advantages of requiring evenly distributed buffer space inside the network and providing jitter-bounded service.

In Section 2, we review some of the issues that arise when designing service disciplines to provide performance guarantees. In Section 3, we describe the algorithm of the proposed RCSP discipline and the corresponding admission control conditions. In Section 4, we present results from simulation experiments on RCSP. In Section 5, the new discipline is compared with the previously proposed Jitter-EDD, Stop-and-Go, and HRR disciplines. We show that the key idea is to separate the functions of rate-control and delay-control; this separation of functions will result in a class of service disciplines, of which RCSP is an instance. Finally, we give conclusions and outline future work in section 6.

2 Background

We consider the paradigm proposed in [9] for providing guaranteed service to clients in a packet switching network: before communication starts, the client specifies its traffic characteristics and performance requirements to the network; the client's traffic and performance parameters are translated into local parameters, and a set of connection admission control conditions are tested at each switch; the new connection is accepted only if its admission would not cause the performance guarantees made to other connections to be violated; during data transfers, each switch will service packets from different connections according to a service discipline; by ensuring that the local performance requirements are met at each switch, the end-to-end performance requirements can be satisfied. Notice that there are two levels of control in this paradigm: connection admission control at the connection level, and service discipline at the packet level. A complete solution needs to specify both the service discipline and the associated connection admission control conditions.

To design a service discipline that guarantees performance, two issues need to be addressed: (1) how to control the interactions between different connections in a single switch; (2) how to control the interactions between different switches.

The first issue relates to the design of a single server: how to allocate delay and bandwidth among different connections. Previously proposed solutions can be summarized as shown in Table 1. As stated before, a multi-level framing strategy introduces dependencies between delay and bandwidth allocation. A sorted priority queue avoids this coupling, but has a higher degree of complexity and may be difficult to implement in very high-speed switches; also, to provide a continuous spectrum of delay bounds, it needs a complicated schedulability test at connection establishment time.

	Bandwidth Allocation	Delay Allocation
Virtual Clock Fair Queueing Delay-EDD	Sorted Priority Queue	
Jitter-EDD	Regulator	Sorted Priority Queue
Stop-and-Go Hierarchical Round Robin	Multi-level Framing	

Table 1: Mechanisms to Allocate Delay and Bandwidth

The second issue concerns the interaction between different switches along a path. A switch can provide local performance guarantees to a connection only when the traffic on that connection is well-behaved. However, network load fluctuations at previous switches may distort the traffic pattern of a connection and cause an instantaneous higher rate at some switch even when the connection satisfies the client-specified rate constraint at the entrance to the network. If the maximum distortion of traffic patterns can be bounded, buffer space can be allocated at each switch to accommodate the distortion. Delay-EDD adopts this approach. Unfortunately, distortions of traffic patterns due to network load fluctuations in the worse case tend to accumulate, and this requires the amount of buffer space allocated to grow linearly along the path. Buffer space can be saved by controlling the distortion of traffic patterns at each switch. In Jitter-EDD, Stop-and-Go, and HRR, traffic patterns are partially or completely reconstructed at each switch so as to offset the effects of network load fluctuation and interaction between switches.

A relevant issue here is that of jitter control. In the literature, there are different definitions of *jitter*. In [12], the term is used to capture the *burstiness* of the traffic, and is defined to be the maximum number of packets in a *jitter averaging interval*. In [6, 18], the term is used to capture the magnitude of the distortion to the traffic pattern caused by the network, and is defined to be the maximum difference between the delays experienced by any two packets on the same connection. In this paper, we call these two quantities *rate jitter* and *delay jitter*, respectively. Eliminating delay jitter at each switch along the path will *completely* reconstruct the traffic pattern at that switch; thus, if the traffic obeys the traffic specification at the entrance to the network, it will obey the specification throughout the network. Controlling the rate jitter at each switch along the path will only *partially* reconstruct the traffic pattern; the partially reconstructed traffic pattern obeys the same traffic specification as the input traffic, but some inter-packet spacing information is lost [17]. Notice that, if the traffic at the entrance to the network has a small rate jitter, controlling delay jitter inside the network will also keep the rate jitter small; however, controlling rate jitter would not automatically eliminate delay jitter. From the network's point of view, the partial reconstruction of traffic patterns, or rate-jitter control inside the network, may be enough to offset the interactions between switches and the network load fluctuations; however, for some applications, a delay-jitter-free *service* is required [6, 7, 18]. If the network does not control delay jitter, applications may have to do buffering at the destination. For this reason, delay jitter control inside the network may be preferred; however, this consideration has to be balanced against the relative complexities of implementing delay-jitter control vs. rate-jitter control.

Although jitter control can save buffer space inside the network and provide bounded-jitter service, it usually requires the service disciplines to be non-work-conserving. Non-work-conserving disciplines may leave the output link idle even when there are packets waiting for transmission; thus, they may not achieve the maximum transmission efficiency of the output link.

3 Rate-controlled Static-Priority Queueing

In this section, we will present a new service discipline called Rate-Controlled Static-Priority (RCSP) queueing. A RCSP server consists of two components: a rate controller and a static priority scheduler, which are responsible for allocating bandwidths and delays to different connections, respectively. The combination of a rate controller and a static priority scheduler decouples the allocation of delay and bandwidth, and also simplifies admission control and implementation.

In this section, we discuss traffic characterization, describe the queueing algorithm, the corresponding admission control policies and the implementation strategy for the non-work-conserving version of RCSP, and finally, present an extension that would allow the algorithm to be work-conserving.

3.1 Traffic Specification

To provide performance guarantees, the network needs to allocate resources on a per-connection basis. This requires clients to specify their traffic characteristics. For this discussion, we use the (X_{min}, X_{ave}, I, P) traffic characterization as suggested in [9, 6], where X_{min} is the minimum packet inter-arrival time, X_{ave} is the average packet inter-arrival time over an averaging interval, I is the length of the interval, and P is the maximum packet size. It should be noted, however, that the algorithm presented here is general and can be easily extended to handle different traffic characterizations such as the (ρ, σ) model proposed in [3].

3.2 Algorithm

A rate-controlled static-priority server has two components: a rate controller and a static-priority scheduler. The rate controller shapes the input traffic from each connection into the desired traffic pattern by assigning an eligibility time to each packet; the scheduler orders the transmission of eligible packets from all the connections.

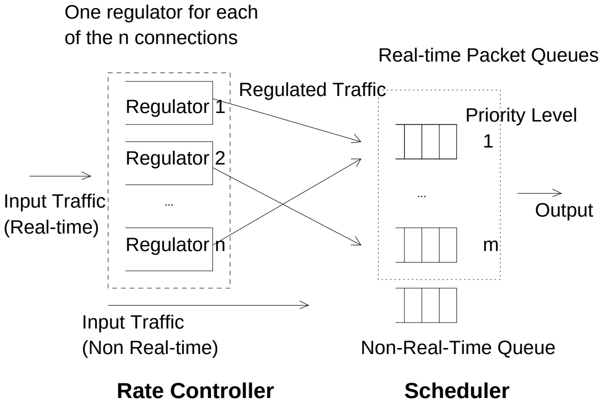

Figure 1: Rate-Controlled Static-Priority Queueing

Conceptually, a rate controller consists of a set of regulators corresponding to each of the connections traversing the switch; each regulator is responsible for shaping the input traffic of the corresponding connection into the desired traffic pattern. Regulators control the interactions between switches and eliminate jitter. We present two types of regulators: (1) *rate-jitter controlling regulator*, which controls rate jitter by partially reconstructing the traffic pattern; (2) *delay-jitter controlling regulator*, which controls delay jitter by fully reconstructing the traffic pattern.

The regulator achieves this control by assigning each packet an eligibility time upon its arrival and holding the packet till that time before handing it to the scheduler. Different ways of calculating the eligibility time of a packet will result in different types of regulators.

For a rate-jitter controlling regulator, ET_j^k, the eligibility time of the $k-th$ packet on a connection that traverses switch j, is defined with reference to the eligibility times of packets arriving earlier at the switch on the same connection:

- $ET_j^1 = AT_j^1$,
- $ET_j^k = \max(ET_j^{k-1} + X_{min} + \tau_j^k, AT_j^k),\ k > 1$

where X_{min} is the minimum packet inter-arrival time for the connection, AT_j^k is the time the $k-th$ packet on the connection arrived at switch j, and τ_j^k is the minimum number such that average packet inter-arrival time in any I period does not exceed X_{ave}. The computation of τ_j^k can be found in [17].

From this definition, we can see that $ET_j^k \geq AT_j^k$ always holds, i.e., a packet is never eligible before its arrival. Also, if we consider the sequence of packet eligibility times at switch j, $\{ET_j^k\}_{k=1,2,\ldots}$, it always satisfies the (X_{min}, X_{ave}, I) traffic characterization.

For a delay-jitter controlling regulator, ET_j^k is defined with reference to the eligibility time of the same packet at the previous switch:

- $ET_0^k = AT_0^k$;
- $ET_j^k = ET_{j-1}^k + d_{j-1} + \pi_{j-1,j}, j > 0$,

where switch 0 is the source of the connection, d_{j-1} is the delay bound, or the maximum waiting time of packets on the same connection at the scheduler of switch $j-1$, and $\pi_{j-1,j}$ is the propagation delay between switch $j-1$ and switch j. (We will show in the next section how the waiting time in a scheduler can be bounded.)

Thus, for a delay-jitter controlling regulator:

- $ET_j^k \geq AT_j^k$, i.e., a packet is never eligible before its arrival;
- $ET_j^{k+1} - ET_j^k = AT_0^{k+1} - AT_0^k$, i.e., the traffic pattern of the connection at the *entrance* to the network is fully reconstructed at the output of the regulator of every switch; if the input traffic obeys the traffic specification at the entrance to the network, it will obey the traffic specification at the output of the regulator of each switch.

Both types of regulators will enforce the traffic specification requirement for each connection so that the traffic going into the scheduler will always satisfy the traffic specification.

The scheduler in a RCSP switch consists of a number of prioritized real-time packet queues and a non real-time packet queue (Figure 1). Packets at priority level 1 have the highest priority. A connection is assigned to a particular priority level at the connection's establishment time; all the packets from the connection will be inserted into the real-time packet queue at that priority level.

As will be discussed in section 3.3, there is a delay bound associated with each priority level; by limiting the number of connections at each priority level using the admission control conditions described in 3.3, the waiting time of each packet at a priority level is guaranteed to be less than the delay bound associated with that level.

The scheduler services packets using a non-preemptive static-priority discipline: which non-preemptively chooses packets in FCFS order from the highest-priority non-empty queue. Non-real-time packets are serviced only when there are no real-time packets; their order is not specified.

The algorithm described above would not provide performance guarantees without connection admission control conditions limiting the number of connections at each priority level. In the next section, we will describe the admission control conditions for RCSP. We will show that, if the admission control conditions are satisfied, by using the RCSP queueing described above, the end-to-end delay, delay jitter, throughput, and loss-free requirements of each admitted connection can be satisfied.

3.3 Admission Control Conditions

The residence time of a packet in a RCSP switch has two components: the *holding* time in the regulator and the *waiting* time in the scheduler (which includes both the queueing time and the service time). In the following, we first consider the case of a single switch; we show that, under certain admission control conditions, the waiting time of a packet is bounded by a fixed delay bound. We then examine the case of two switches connected in cascade; we show that, if the connection admission control conditions are satisfied at both switches, the regulator of the second switch might delay a packet only if the packet is *ahead* of schedule in the first switch; the sum of the waiting time of a packet in the first switch and the holding time in the second switch is no greater than the delay bound of the first switch. From this result, we establish the end-to-end delay and delay-jitter characteristics of a connection, and the buffer space requirement at each switch.

The following theorem gives the conditions to bound the waiting time of packets at each priority level in the scheduler.

Theorem 1 *Let* $d_1, d_2, \ldots, d_n$ $(d_1 < d_2 < \cdots < d_n)$ *be the delay bounds associated with each of the* n *priority levels, respectively, in an RCSP switch. Assume that the* j^{th} *connection among the* i_k *connections traversing the switch at priority level* k *has the traffic specification* $(Xmin_{k,j}, Xave_{k,j}, I_{k,j}, P_{k,j})$*. Also assume that the link speed is* l*, and the size of the largest packet that can be transmitted onto the link is* P_{max} *. If*

$$\sum_{k=1}^{m} \sum_{j=1}^{i_k} \lceil \frac{d_m}{Xmin_{k,j}} \rceil P_{k,j} + P_{max} \leq d_m l \ , \quad (1)$$

the waiting time of an eligible packet at level m *is bounded by* d_m.

The longest waiting time in the scheduler for a level-m packet corresponds to the case in which a lower-priority packet is being transmitted when the packet arrives at the scheduler, and is followed immediately by the longest possible transmission of packets with higher or equal priorities.

Since the length of the lower-priority packet is bounded by P_{max}, inequality (1) ensures that the level-m packet will be transmitted within d_m units of time after its arrival even in the worst case.

Theorem 1 gives the admission control condition for switch k. For each priority level m, a state variable $MaxBitsInD_m$ is kept and is initialized to be P_{max}. When an establishment request for a new connection with traffic specification $(Xmin, Xave, I, P)$ comes in, the following tests are performed from priority level 1 to n:

$$MaxBitsInD_m + \lceil \frac{d_m}{Xmin} \rceil P \leq d_m l \quad (2)$$

The new connection can be placed into priority level k without jeopardizing the performance guarantees of other existing connections if and only if equality (2) holds for $k, k+1, \cdots, n$. Usually, there are multiple priority levels that the new connection can be placed into. Different connection establishment schemes can be used to choose the priority level for the new connection. One scheme for establishing a real-time connection is described in [9], in which a new connection is setup with a one-round-trip message: during the forward trip of the establishment message, the new connection is assigned the smallest possible delay bound at each switch to maximize the probability that the new connection's end-to-end delay bound can be met; during the return trip, the local delay bound at each switch can be relaxed if it turns out that the offered delay bound is tighter than the requested one. This is a greedy delay allocation policy: although it minimizes the blocking probability of the new connection, it increases the blocking probability for other connection establishment requests before the delay bound for the first connection is relaxed. How to assign the local delay bound, or priority level, to a new connection request during the forward trip is a *policy* issue, which deserves further research. If we decide to place the new connection in priority level h, state variables need to be updated accordingly for $m = h, \cdots, n$:

$$MaxBitsInD_m \Longleftarrow MaxBitsInD_m + \lceil \frac{d_m}{Xmin} \rceil P$$

The admission control conditions in Theorem 1 avoid both bandwidth and scheduling saturation. The tests have just $O(n)$ steps, where n is the number of priority levels, and are simple enough to be used in a fast connection establishment scheme [16].

Theorem 1 just gives the bounds for the waiting times of packets in the scheduler; the holding times of packets in the regulators have not been taken into account. In the previous section, we have introduced two types of regulators: the rate-jitter controlling regulator and the delay-jitter controlling regulator. In the following lemma, we establish the delay characteristics of two switches connected in cascade. when each type of regulator is used.

Lemma 1 *Consider a connection with traffic specification $(Xmin, Xave, I, P)$ passing through two RCSP switches connected in cascade. Assume that the connection is assigned to the priority level with delay bound d at the first switch. Furthermore, assume that the admission control policy as defined in Theorem 1 is satisfied at both switches. It follows that, for any packet on the connection, (1) the waiting time in the first switch plus the holding time in the second switch is less than d if a rate-jitter controlling regulator is used in the second switch; (2) the waiting time in the first switch plus the holding time in the second switch equals d if a delay-jitter controlling regulator is used at the second switch.*

The key observation for understanding the lemma is that a packet is held in a regulator only when the packet was transmitted ahead of schedule by the previous switch; the amount of holding time in the regulator is never greater than the amount of time the packet is ahead of schedule. When a rate-jitter controlling regulator is used, the amount of time a packet is to be held is calculated according to the packet spacing requirement, which may be less than the amount of time it was ahead of schedule at the previous switch. When a delay-jitter controlling regulator is used, the amount of holding time is exactly the amount of time the packet was ahead of schedule when it left the previous scheduler.

The following theorem establishes the conditions for bounding the end-to-end delay for a connection.

Theorem 2 *Consider a connection with traffic specification (X_{min}, X_{ave}, I, P) passing through k switches connected in cascade where the end-to-end propagation delay is Π. Assume that the connection is assigned to the priority levels with delay bounds $d_{i_1}, \ldots, d_{i_k}$ at each switch, respectively. Furthermore, assume that the admission control policy defined in Theorem 1 is satisfied at all the switches. If the traffic on the connection obeys the (X_{min}, X_{ave}, I, P) specification at the entrance to the first switch, (1) the end-to-end delay for any packet on the connection is bounded by $\sum_{j=1}^{k} d_{i_j} + \Pi$ if rate-jitter controlling regulators are used; (2) the end-to-end delay and delay jitter for any packet are bounded by $\sum_{j=1}^{k} d_{i_j} + \Pi$ and d_{i_k}, respectively, if delay-jitter controlling regulators are used; (3) amount of buffer space equal to $(\lceil \frac{d_{i_{j-1}}}{X_{min}} \rceil + \lceil \frac{d_{i_j}}{X_{min}} \rceil)P$ is needed by connection i at switch j to prevent packet loss ($j = 1, \ldots, k$; $d_{i_0} = 0$).*

For the first two parts of the theorem, consider the end-to-end delay of one packet in the connection, which can be expressed as:

$$\sum_{j=1}^{k} (h_j + w_j) + \Pi$$

where h_j and w_j are the holding and the waiting times at switch j. If we re-arrange the terms, the formula becomes:

$$h_1 + \sum_{j=1}^{k-1}(w_j + h_{j+1}) + w_k + \Pi$$

If the traffic obeys the (X_{min}, X_{ave}, I, P) characterization at the entrance to the first switch, there is no holding time in the first regulator, so $h_1 = 0$. From Lemma 1, $w_j + h_{j+1} = d_{i_j}$ holds for a delay-jitter controlling regulator and $w_j + h_{j+1} \leq d_{i_j}$ holds for a rate-jitter controlling regulator; furthermore, $w_k \leq d_{i_k}$ holds in both cases; the correctness of the first two parts of the theorem immediately follows.

To verify the third part of the theorem, notice that the longest times a packet can stay in the regulator and the scheduler of the $j-th$ switch are $d_{i_{j-1}}$ and d_{i_j}, respectively; since the minimum packet inter-arrival time is $Xmin$, it follows that the maximum numbers of packets in the regulator and the scheduler are $\lceil \frac{d_{i_{j-1}}}{X_{min}} \rceil$ and $\lceil \frac{d_{i_j}}{X_{min}} \rceil$, respectively.

3.4 Implementation

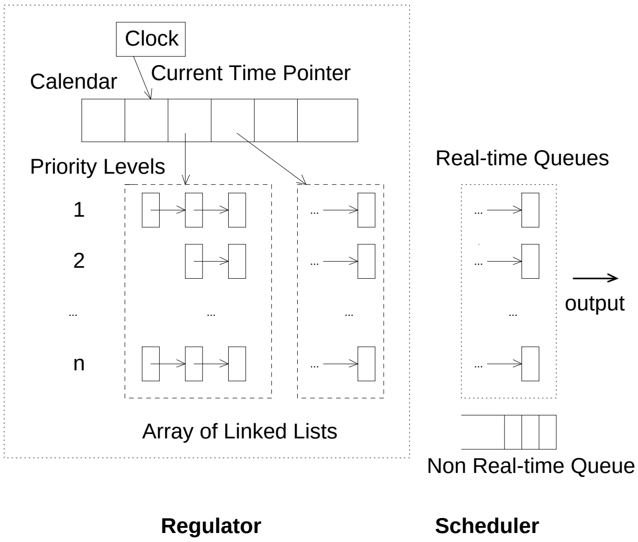

Figure 2: Implementation of RCSP

In this section, we present one implementation of the RCSP queueing discipline. We believe that this implementation is simple enough to run at very high speed.

We have shown in Section 3.2 that a RCSP server has two components, a scheduler and a rate controller. The scheduler consists of multiple prioritized FCFS queues, and the rate controller consists of a set of regulators corresponding to each connection. Notice that the conceptual decomposition of the rate controller into a set of regulators does not imply that there must be multiple physical regulators in an implementation; a common mechanism can be shared by all logical regulators. Each regulator has two functions: computing the eligibility times for incoming packets on the corresponding connection, and holding packets till they become eligible. Eligibility times for packets from different connections are computed using the same formula (as described in Section 3.2) with different parameters; holding packets is equivalent to managing a set of timers; the mechanism for managing timers, which is a calendar queue [2, 17], can be shared by all regulators.

Figure 2 shows the proposed implementation. Each of the real-time queues is implemented as a linked list. The operations performed on a linked list are: deleting the packet at the head (when the next packet is chosen for transmission) and appending a packet or linked list of packets at the tail (when a packet arrives or the clock ticks; see below).

The rate controller is implemented using a modified version of a calendar queue. A calendar queue consists of a clock and a calendar, which is a pointer array indexed by time; each entry in the calendar points to an array of linked lists indexed by priority levels. The clock ticks at fixed time intervals. Upon every tick of the clock, the linked lists in the array indexed by the current time are appended at the end of the scheduler's linked lists: packets from the linked list of one priority level in the rate-controller are appended to the linked list of the same priority level in the scheduler.

Upon the arrival of each packet, the eligibility time of the packet, ET, is calculated; if $\lfloor \frac{ET}{Tick} \rfloor$ is equal to the current clock time, where $Tick$ is the clock tick interval, the packet is appended at the end of the corresponding real-time queue of the scheduler; otherwise, the packet is appended at the corresponding linked list at the calendar queue entry indexed by $\lfloor \frac{ET}{Tick} \rfloor$.

Notice that, in this implementation, packets may become eligible up to $Tick$ time units earlier than their actual eligibility times. The admission control conditions need to be modified to accommodate for this; inequality (1) in Theorem 1 should become:

$$\sum_{k=1}^{m}\sum_{j=1}^{i_k}\lceil \frac{d_k + Tick}{Xmin_{k,j}} \rceil P_{k,j} + P_{max} \leq d_m l$$

$d_k + Tick$ is used instead of d_k in the numerator to account for rounding errors in eligibility time due to the use of a calendar queue. Similar changes need to be made to Theorem 2: the buffer space requirement should be $(\lceil \frac{d_{i_{j-1}}+Tick}{X_{min}} \rceil + \lceil \frac{d_{i_j}}{X_{min}} \rceil)P$; for a delay-jitter controlled connection, the end-to-end delay jitter bound is $d_{i_k} + Tick$. A smaller $Tick$ would introduce less rounding error, but would have a larger overhead. The effect of the value of $Tick$ on performance should be studied further.

As can be seen, the data structures used in the proposed implementation are simple: arrays and linked lists; the operations are all constant-time ones: insertion at the tail of a linked list and deletion from the head of a linked list. We believe that it is feasible to implement this in a very high speed switch.

3.5 Work-Conserving Version of RCSP

The RCSP policy presented before is non-work-conserving — it may leave the server idle even when there are packets waiting for transmission. Although non-work-conserving algorithms have the advantage of requiring uniformly distributed buffer space inside the network and being able to provide bounded-jitter service, they may not achieve the maximum transmission efficiency of the output link. In this section, we present a work-conserving version of RCSP.

In the work-conserving RCSP, there is one more queue in the scheduler, called the *stand-by queue*. It works as follows:

- All the packets in the rate-controller are also queued in the stand-by queue. Packets are inserted or deleted from the rate-controller and the stand-by queue simultaneously.

- The scheduler will service the next packet in the stand-by queue only if there are no non-real-time packets and eligible real-time packets in the scheduler.

The stand-by queue allows the non-eligible packets to *stand by* at the scheduler, so that they can be transmitted when there is spare capacity at the output link. Similar ideas have been independently proposed in [1, 4].

Notice that with the introduction of the stand-by queue, the service discipline becomes work-conserving. It can be easily shown that Theorem 1 and Theorem 2 still hold for the work-conserving RCSP with exception to the statements on delay-jitter and buffer space. Since ineligible packets can be transmitted via the stand-by queue, the traffic pattern of a connection is no longer preserved at each switch. In order to accommodate the potential distortion to the traffic pattern, more buffer space needs to be reserved in the downstream switches; at switch i, the buffer space requirement is $(\lceil \frac{\sum_{k=1}^{j-1} d_{i_k} + Tick}{X_{min}} \rceil + \lceil \frac{d_{i_j}}{X_{min}} \rceil) P$.

4 Simulation Experiments

The network being simulated is shown in Figure 3. All the links are 10 Mbps.

The first experiment compares the effects of rate-jitter control and delay-jitter control. The channel being measured traverses nodes (1, 3, 5, 6) with a guaranteed end-to-end delay bound of 48 ms. Additional channels that